AMENDMENTS TO THE SPECIFICATION:

Please amend the specification as indicated on the attached substitute specification with markings. Applicant also attaches a clean version of the substitute specification. The undersigned believes that no new matter has been added, but the Examiner is encouraged to carefully review the "mark on" version of the revised proposed substitute specification and raise any possible "new matter" issues he may see.

SYSTEM AND METHOD FOR ESTABLISHING SECURE COMMUNICATION

FIELD

[0001] The technology herein relates to encoding systems and protocols, and more specifically to systems and protocols for establishing a shared secret key for two-way secure communications.

BACKGROUND AND SUMMARY

[0002] When two parties wish to communicate securely, an efficient mechanism for doing so is the use of a shared secret session key, i.e., a key known only to the two parties that can be used symmetrically to both encrypt and decrypt messages between them for the duration of a communication session. Various methods exist to achieve this, and each have advantages and disadvantages.

[0003] Parties can use a trusted key authority that distributes the shared secret key to each of them separately using their unique key encryption key. However, this technique requires the storage of keys--i.e., it is not portable with a user, and if a key encryption key is compromised the system loses its integrity and past communications can be decrypted.

[0004] The "Diffie-Hellman" technique, described in U.S. Pat. No. 4,200,770, permits generation of a shared secret key without the use of encryption. Each party generates a large random number. By way of example, party A generates the number X and party B generates the number Y. Each party sends its number through a particular kind of one way function and transmits the output. Only knowledge of one number (X or Y), and the value of the other number sent through the one-way function is sufficient to generate the shared secret key. A drawback of the Diffie-Hellman technique is that each side uses a non-shared random number (X or Y) in independently generating the shared secret key. A result of the use of non-shared random numbers is that each side performs large

exponential and modulo calculations when performing one-way functions and generating the shared secret key, resulting in a high computational load on both sides. These calculations are required in order to make it computationally infeasible for an eavesdropper to combine the two-shared numbers in order to obtain the shared secret key.

[0005] Variations on the Diffie-Hellman technique exist that attempt to make it more secure. These variations suffer from the same computational burden as the standard Diffie-Hellman technique. For example, U.S. Pat. No. 5,953,424 describes a system with a modification of the key generation technique. In addition to the usual Diffie-Hellman computations on the original numbers (X and Y) and the transmitted numbers (which result from calculations), the '424 patent describes extra factors that may be combined with the standard Diffie-Hellman factors. These factors are not transmitted, so they must be knowable in advance by the communicating parties.

[0006] Another variation of the Diffie-Hellman technique is disclosed in U.S. Pat. No. 5,440,635. In this variant, the transmitted numbers are further encrypted using a symmetric key cryptosystem before being transmitted. This doesn't ameliorate any of the disadvantages of Diffie-Hellman, noted above.

[0007] A message exchange technique employing a combination of public and private key cryptography to communicate a secret key from one party to another is described in U.S. Pat. No. 5,241,599. The '599 patent requires that each party share knowledge of a secret. A calling party generates a random public key/private key pair, and communicates the public key to the called party using their shared secret. The called party then communicates the secret key to the calling party using both the public key and the shared secret. The technique of the '599 patent suffers from several limitations. The calling party must generate a random public key/private key pair, which is a costly computation that is often preferably performed by the called party. Also, the secret key may be compromised in advance by manipulating the called party to affect the secret key

it uses or computes. No manipulation or compromise of the calling party is required.

[0008] What is needed is a system and method for establishing secure communication.

[0009] The exemplary illustrative non-limiting technology herein provides an encoding protocol for communicating parties to each obtain a shared secret key.

[0010] One advantage of the exemplary illustrative non-limiting technology herein is that it is less computationally intensive than previous cryptographic systems to obtain a shared secret key.

[0011] Another advantage of the exemplary illustrative non-limiting technology herein is that the calling party is not required to perform any large computations.

[0012] Yet another advantage of the exemplary illustrative non-limiting technology herein is that it is highly resistant to attacks, including eavesdropping, impersonating a party, replay attacks, tampering with or probing a party before or after a communications session, and password database hijacking.

[0013] Still another advantage of the exemplary illustrative non-limiting technology herein is that it can be used either with or without certificates or physical tokens such as smart cards or biometric devices.

[0014] In an exemplary illustrative non-limiting method for obtaining a shared secret key, a party identifies a first shared random number and a second shared random number, and obtains the shared secret key from an output of a combining function having a first input including the first shared random number and having a second input including the second shared random number.

[0015] In a further aspect of the exemplary illustrative non-limiting implementation , the shared secret key is used to transform messages.

[0016] In another exemplary illustrative non-limiting implementation, a party encodes a first shared random number, decodes a second shared random number, and obtains the shared secret key from an output of a combining function having a first input including the first shared random number and having a second input including the second shared random number.

[0017] In a further exemplary illustrative non-limiting implementation, a party encodes a first shared random number and a second key using a first key obtained using information obtained from a password; decodes a second shared random number using a third key; and obtains the shared secret key from an output of a combining function having a first input including the first shared random number and having a second input including the second shared random number.

[0018] In a still further exemplary illustrative non-limiting implementation, the second key and the third key form an asymmetric key pair.

[0019] In another exemplary illustrative non-limiting implementation, a party decodes a first shared random number, encodes a second shared random number, and obtains the shared secret key from an output of a combining function having a first input including the first shared random number and having a second input including the second shared random number.

[0020] In a further exemplary illustrative non-limiting implementation, a party decodes a first shared random number and a second key using a first key obtained from information obtained from a password, encodes a second shared random number using the second key, and obtains the shared secret key from an output of a combining function having a first input including the first shared random number and having a second input including the second shared random number.

[0021] In another exemplary illustrative non-limiting implementation, a party communicates a first shared random number and a second shared random number, and obtains the shared secret key from an output of a combining function

having a first input including the first shared random number and having a second input including the second shared random number.

[0022] In a further exemplary illustrative non-limiting implementation, the party communicates an asymmetric key and a timestamp with the first shared random number, and a timestamp with the second shared random number.

[0023] In still another exemplary illustrative non-limiting implementation, a device including at least one processor executes software instructions identifying a first shared random number and a second shared random number, and obtains the shared secret key from an output of a combining function having a first input including said first shared random number and having a second input including said second shared random number.

[0024] In yet another exemplary illustrative non-limiting implementation, a machine-readable storage medium contains instructions for a processor, including encoded computer means for identifying a first random number, encoded computer means for identifying a second random number, and encoded computer means for obtaining the shared secret key from an output of a combining function having a first input including said first shared random number and having a second input including said second shared random number.

BRIEF DESCRIPTION OF THE DRAWINGS

[0025] These and other features and advantages will be better and more completely understood by referring to the following detailed description of exemplary non-limiting illustrative implementations in conjunction with the drawings of which:

[0026] FIG. 1a illustrates an exemplary illustrative non-limiting peer-to-peer embodiment , including various exemplary non-limiting implementations of each peer.

[0027] FIG. 1b illustrates an exemplary illustrative non-limiting client-server embodiment , including various embodiments of a client.

[0028] FIG. 2 illustrates a sequence of exemplary illustrative non-limiting operations performed by a peer, a client, or a server.

[0029] FIG. 3 illustrates an exemplary illustrative non-limiting sequence of operations performed by a calling party in a peer-to-peer embodiment, or a client in a client-server embodiment .

[0030] FIG. 4 illustrates a sequence of exemplary illustrative non-limiting operations performed by a called party in a peer-to-peer embodiment, or a server in a client-server embodiment.

[0031] FIG. 5 illustrates a packet that may be used in an exemplary illustrative non-limiting signal embodiment .

[0032] FIG. 6 illustrates a sequence of actions and communications among a user, a calling party, and a called party in one exemplary illustrative non-limiting embodiment .

DETAILED DESCRIPTION

[0033] The exemplary illustrative non-limiting technology herein provides a system and method for two parties to determine a shared secret key. Each party identifies a first shared random number and a second shared random number. Then, a combining function is used to obtain a shared secret key using the first shared random number and the second shared random number. Hence, knowledge of the combining function coupled with knowledge of the first shared random number and the second shared random number is sufficient to obtain the shared secret key. It is preferred, therefore, for the shared random numbers to be communicated between the parties without permitting would-be attackers to learn both shared random numbers. In some exemplary illustrative non-limiting embodiments, this is accomplished by using a password. Specifically, each party

has access to either information obtained from a password, or of the password itself. In an exemplary illustrative non-limiting embodiment, the password is associated with a user.

[0034] A shared secret key may be used along with a symmetric encryption system such as IDEA (International Data Encryption Algorithm) in order to efficiently and securely perform communications between two parties. It is desirable that the shared secret key be secure and that the two parties authenticate each other, while minimizing the computational load necessary to obtain the shared secret key.

[0035] The system and method of the exemplary illustrative non-limiting technology herein can be implemented in a local area network, wide area network, public access network (e.g., Internet), in a network of networks such as a hybrid network, or in other communication environments as would be apparent. In a network implementation that conforms to the OSI model, the shared secret key may be viewed as a session key. In an OSI-compliant exemplary illustrative non-limiting embodiment, the protocol for obtaining a shared secret key resides at layers 5 and/or 6 of the OSI model, and supports secret key based encoded transport of any developer-defined or selected client-server or peer-to-peer protocol. In addition, a user authentication graphical user interface (GUI) for any developer-defined applications may be provided above the exemplary illustrative non-limiting protocol, as would be apparent. In addition, the exemplary illustrative non-limiting protocol may be encapsulated within another protocol such as a firewall, a virtual private network, or both, or any other protocol as would be apparent. With encapsulation, an application benefiting from the authentication and encoded transport of the exemplary illustrative non-limiting implementation does not need to be aware of the details or even the existence of the underlying protocol.

[0036] In generating a shared secret key according to some exemplary illustrative non-limiting embodiments, the system and method provides a secure

and robust user authentication protocol, based on asymmetric key encoding. In an exemplary illustrative non-limiting embodiment, a first party communicates a public key to a second party using information obtained from a password associated with a user of the second party. The first and second parties exchange shared secret keys which no outside observer can obtain without information about the password and information about a private key associated with the public key, which is not communicated.

[0037] FIG. 1a and FIG. 1b depict exemplary illustrative non-limiting embodiments of the environment. FIG. 1a shows an exemplary peer-to-peer embodiment. Users 101a through 101e and 105a through 105e are graphically depicted as human users for ease of representation, although they may be automated entities such as applications, daemons, or other electronically-driven users. A user need not be associated with a particular human if it is a non-human user such as a software-driven user, although it can be. Calling users 101 communicate with (i.e., are connected to) exemplary calling platforms 102a through 102d, and called users 105 communicate with exemplary called platforms 104a through 104d.

[0038] The term `party` may refer to a user, a platform, or both a user and a platform with which the user is associated. In one exemplary illustrative non-limiting embodiment, a calling party is a party that initiates a communications session; and a called party is a party that responds to a calling party's initiation of a communications session. Network 103 represents any network or network of networks, such as the Internet, and may include firewalls, virtual private networks, and any kinds of connections (e.g., wireless, Ethernet, etc.), routers, gateways, protocols, and other components as would be apparent. Exemplary platforms 102a and 104a represent laptop computers, which may be connected to network 103 via wire-line or wireless connections, as would be apparent. Exemplary platforms 102b and 104b represent workstations or personal computers. Exemplary platforms 102c and 104c represent handheld devices. Exemplary platforms 102d

and 104d represent servers, which may be better equipped than the other exemplary platforms to handle simultaneous connections to multiple users including remote users, as illustrated. Other platforms may be used, such as a dumb terminal, as would be apparent. For the purposes of this specification, a platform may comprise one or more pieces of hardware, a process running on the hardware, a process in memory or storage, or a combination thereof. For example, platform 104d may represent the physical server, i.e., the hardware comprising the server. Alternatively, platform 104d may represent a server process running on the server or in memory on the server. As an additional alternative, platform 104d may represent a combination of these things. A non-human user may be running at least partially on a platform with which it is associated, or it may be running completely remotely. The meaning of platform varies throughout the specification, and is in each case subject to any of the interpretations just enumerated.

[0039] FIG. 1a depicts a peer-to-peer configuration, in which a party (possibly including one of exemplary users 101a through 101e and/or one of corresponding exemplary platforms 102a through 102d) may be either a calling party or a called party. Similarly, any of exemplary users 105a through 105e and/or a corresponding exemplary platform 104a through 104d, could be either a calling or a called party. FIG. 1b represents a client-server configuration in which a client (possibly including one of exemplary users 101a through 101e and/or one of corresponding exemplary platforms 102a through 102d) is the calling party and a party on the server side, including user 105d and/or server platform 104d is the called party 105d. The exemplary illustrative non-limiting technology herein may be used to connect parties, including a calling party and a called party to provide for authentication and/or secure communication between the calling party and the called party, through network 103.

[0040] A random number has the property that it is difficult for a would-be attacker to determine the random number without obtaining information characterizing the random number. A random number that is either communicated

or intended to be communicated between a calling party and a called party is referred to as a shared random number. In order to communicate a shared random number, a party may send or receive the shared random number or information from which the shared random number may be obtained without additional information specific to the shared random number and without an astronomical number of computations. A shared random number may be sent or received by transmitting or receiving over a communications channel or a network information from which the shared random number may be easily inferred, such as the shared random number itself.

[0041] The exemplary illustrative non-limiting technology herein employs at least two shared random numbers to obtain a shared secret key. The shared random numbers are passed through a combining function to obtain information used to determine the shared secret key. A combining function may be any function, including a compound function, i.e., a function of functions, that has a first input including a first number and a second input including a second number. Additionally, a combining function has the property that knowledge of a single input to the combining function does not substantially reduce the difficulty of determining an output of the combining function. It is preferred that a combining function be used such that knowledge of a single input to the combining function does not reduce the difficulty of determining an output of the combining function at all.

[0042] A benefit of passing shared random numbers through a combining function to obtain a shared secret key is that a would-be attacker will be unable to compromise the shared secret key by obtaining only one random number, either through observation or manipulation of a party or its communications, even with knowledge of the protocols used and the combining function itself. A combining function may involve one or more of a number of operations on its inputs and any parameters that might be inherent to the combining function. For example, addition, subtraction, multiplication, division, exponentiation, logarithms,

trigonometric functions, and/or modulo operations could be used. A combining function could include a logical function, such as OR, AND, NOR, NAND, XOR, and/or XNOR, blending or merging (scaling, and then addition), bit shifting, concatenation, truncation, or any combinations thereof. Additional operations, conditional operations, and various combinations of operations, in serial and/or parallel may be used, as would be apparent. In an exemplary illustrative non-limiting embodiment, an XOR operation is used, which would permit the use of a combining function as simple as specifying an output of a combining function to be equal to an XOR of two inputs.

[0043] FIG. 2 illustrates a set of steps performed by a party in an exemplary illustrative non-limiting embodiment . In step 201, the party shares a first random number R1. The random number could be shared by being communicated to or from another party, possibly through intermediaries. In step 202, the party shares a second random number R2. In step 203, the party obtains an output K of a combining function f() with a first input including R1 and a second input including R2. In FIGS. 2-4, for ease of presentation, an exemplary illustrative non-limiting embodiment of the combining function is shown with two inputs consisting of R1 and R2, respectively, yielding K=f(R1,R2).

[0044] FIG. 3 illustrates a set of steps performed by a calling party in an exemplary illustrative non-limiting embodiment. In an exemplary illustrative non-limiting client-server embodiment, FIG. 3 illustrates a set of steps performed by a client. For ease of description, the party executing the steps in FIG. 3 will be referred to as a `calling party,` which encompasses the term `client.` In step 301, the calling party receives a first shared random number R1. In step 302, the calling party sends a second shared random number R2. In step 203, the calling party obtains an output K of a combining function f( ) with a first input including R1 and a second input including R2.

[0045] FIG. 4 illustrates a set of steps performed by a called party in an exemplary non-limiting embodiment . In an exemplary client-server embodiment,

FIG. 4 illustrates a set of steps performed by a server. For ease of description, the party executing the steps in FIG. 4 will be referred to as a `called party,` which encompasses the term `server.` In step 401, the called party sends a first shared random number R1. In step 402, the called party sends a second shared random number R2. In step 203, the called party obtains an output K of a combining function f() with a first input including R1 and a second input including R2.

[0046] When a party communicates information, it may do so over network 103. In order to send information, the information may be encoded to comply with a communications protocol. It may further be encoded with keys, passwords, or information derived therefrom. Encoding with keys or passwords or information derived therefrom may comprise encryption, or other forms of transforming data as would be apparent. If information is sent through network 103, it may be encoded in packets, such as IP packets. An exemplary IP packet is illustrated in FIG. 5. The IP packet of FIG. 5 includes an IP header 501 including a source address 502 and a destination address 503, and embedded data 504 including part of the information sent through the network. When a party receives information through network 103, it may decode information according to a network protocol, such as the Internet Protocol. Information may be decoded with keys or passwords or information derived therefrom. Decoding with keys or passwords or information derived therefrom may comprise decryption. Decoding of information may also comprise parsing, in which information is extracted from data passed through a network, such as a packet or packets.

[0047] Information communicated over network 103 is propagated through network 103 between communicating parties. Networks may comprise many kinds of links, including wireless links. Signals sent over the network may be embedded in a carrier wave, and may be propagated, e.g., as an analog or a digital signal.

[0048] If keys or passwords, including information derived from them are used to encode or decode information, then a key or password may itself be encoded. In one exemplary illustrative non-limiting embodiment, a password is

encoded to obtain a 128-bit key. One method of performing this encoding is as follows. In a first step, lcase.oval-hollow. is used to transform all alphabetic characters to lower-case. In a second step, characters are transformed based on a stored table. In a third step, all bits in the transformed representation of each character are concatenated. In a fourth step, the resulting bit stream is repeated until the total bit length is 128. In a fifth step, IDEA is used to encrypt the original password using the 128-bit stream as the secret key. In a sixth step, the cyphertext is padded to create a 128-bit key, which may be used to encode and/or decode information.

[0049] The above method for encoding a password is an example of using a one-way function. A one-way function has the property that it is computationally infeasible to determine an input to the function by using an output of the function. The six-step method for encoding the password is equivalent to passing the password through a function. In this case, it is a one-way function because the encrypting step makes it computationally infeasible to recover the input by using the output. Another example of a one-way function is a hash.

[0050] FIG. 6 illustrates an exemplary illustrative non-limiting embodiment. The four columns of the figure represent users and platforms. Specifically, column 101 represents a calling party, column 102 represents a calling platform, column 104 represents a called platform, and column 105 represents a called user. Calling user 101 is connected to calling platform 102, and called user 105 is connected to called platform 104. Arrows represent messages sent between the four entities (columns) in FIG. 6, and numbers without arrows represent operations that may be performed by either or both of the adjacent columns. For example, step 601 represents a message sent from calling user 101 to calling platform 102, and step 604 represents a step that may be performed by called platform 104, called user 105, or both.

[0051] In step 601, calling user 101 sends user identification information (User ID) and a password to calling platform 102. The password does not need to

be placed in storage on calling platform 102, and could be held in memory on platform 102 just long enough for it to fulfill its use. If calling user 101 is a human user, the action of sending the User ID and password to platform 102 could be triggered by the action of the human user typing in the User ID and password on a keyboard or other alphanumeric input device. In step 602, calling platform 102 sends information to called platform 104 including information obtained from the User ID and information concerning a protocol and version that calling platform 102 is capable of using.

[0052] Step 603 is carried out by the calling party, which may comprise calling user 101, calling platform 102, or both. In step 603, a first key, denoted K_{user}, is generated using information obtained from the password. In some exemplary illustrative non-limiting embodiments, the first key is generated as an output of a one way function having an input including the password. In other embodiments, other methods of encoding the password may be used to obtain K_{user}. Other methods may be used to obtain K_{user}, as would be apparent.

[0053] Steps 604 through 606 are carried out by the called party, which may comprise called platform 104, called user 105, or both. In step 604, the called party identifies a first shared random number R1. This and other identification steps could comprise, for example, generating R1, looking up R1 in a table, obtaining R1 from an external source, and/or other methods as would be apparent. In step 605, the called party identifies an asymmetric key pair comprising two corresponding asymmetric keys, e.g., a public key and a private key. For ease of representation, the two asymmetric keys are denoted K_{public} and $K_{private}$.

[0054] In step 606, a first key, K_{user}, is obtained by using information obtained from the User ID. In some client-server embodiments, the first key is obtained by performing a table lookup using information obtained from the User ID. For example, one embodiment that uses a first key comprising an encoded password requires the called party to have access to a table of passwords, which

are indexed by User ID. In this way, step 606 may be performed by looking up the password in the password table using the User ID or information obtained therefrom, and sending the password through a one-way function to obtain the first key. Alternatively, the password table may contain encrypted or encoded passwords, and a simple table lookup may be used to obtain the first key without an extra encoding step. In another embodiment, the password table is itself encoded. The table could be encoded as a whole, or it could be broken into sub-tables or fields, with each sub-table or field encoded separately. If the password table is encoded, the called party may decode the table or a relevant part of it as part of step 606.

[0055] Step 606 may be performed in other ways. For example, in some embodiments including a peer-to-peer embodiment, information obtained from the User ID is used to obtain from a trusted third party either the first key or information used to generate the first key.

[0056] In step 607, the called party sends the calling party a first message encoded with the first key, K_{user}. The first message includes the first shared random number, R1, and one of the two asymmetric keys identified in step 605, which we arbitrarily denote K_{public}. The first message also includes a timestamp in some embodiments, although in a preferred embodiment timestamps are not used because of timing synchronization issues. The first message may be denoted K_{user} (R1,K_{public},timestamp).

[0057] Steps 608 and 609 are performed by the calling party, comprising the calling user 101 and/or the calling platform 102. In step 608, the first message is decoded. In order to decode the first message, the calling party uses K_{user}. Then, R1 and K_{public} are obtained from the first message. One method to obtain R1 and K_{public} from the first message is to parse the message, including any header information that may exist. If the first message includes a timestamp, then it may

be obtained from the first message and compared to the actual time. In step 609, a second shared random number R2 is identified.

[0058] In step 610, the calling party sends the called party a second message encoded with the asymmetric key K_{public}. The second message contains the second shared random number R2. The second message also includes a timestamp in some embodiments, although in a preferred embodiment timestamps are not used because of time synchronization issues. The second message may be denoted K_{public} (R2,timestamp).

[0059] Step 611 is carried out by the called party. In step 611, the second message is decoded using the other asymmetric key, $K_{private}$, to obtain R2. One method to obtain R2 from the second message is to parse the message, including any header information that may exist. If the second message includes a timestamp, then it may be obtained from the second message and compared to the actual time.

[0060] Step 612 is carried out both by the calling party and by the called party. In step 612, the shared secret key, K_{ss}, is obtained from an output of a combining function having a first input including the first shared random number and a second input including the second shared random number. In one embodiment, the shared secret key is the output of a combining function having two inputs consisting of the two shared random numbers. This is denoted by $K_{ss}=f(R1,R2)$.

[0061] In step 613, the two parties communicate using K_{ss}. K_{ss} may be used, e.g., in any symmetric encryption system to transform messages. A sender of a message transforms the message by encoding it using K_{ss}, and a receiver of a message transforms the message by decoding it using K_{ss}.

[0062] The exemplary illustrative non-limiting technology herein may be used with client/server and peer-to-peer embodiments. In a peer-to-peer architecture, the called party could also pre-generate keys. A problem with peer-

to-peer is that password management becomes difficult because there is no centralized repository. Because password information is extremely sensitive, passing passwords becomes a challenge.

[0063] Use of exemplary illustrative non-limiting implementations herein accrues numerous benefits. One benefit is that there is no need to use certificates. Therefore, there is no need to register with Certificate Authorities and keep them up to date. This removes an external source of unnecessary problems--a Certificate Authority's errors. Furthermore, if a user wishes to log in to a server using a client-server embodiment , then any computer logged in to (i.e., used as a client) can immediately have access without ensuring that the computer has been registered with a certificate authority.

[0064] Another benefit of the exemplary illustrative non-limiting technology herein is that in some embodiments keys are not stored on local computers. In embodiments in which keys are generated dynamically, keys never need to be stored on a filesystem. Storing keys on the system allows someone who is capable of compromising a computer to steal the keys, and decrypt messages past, present, or future. In some exemplary illustrative non-limiting embodiments , keys are constantly changed and never stored on a computer's hard drive.

[0065] Yet another benefit of the exemplary illustrative non-limiting implementations is that it is not necessary for a calling party to generate keys. Generating asymmetric key pairs can be computationally expensive, resulting in system delays. A calling party may have access only to limited computational resources. If an asymmetric key pair is generated on a limited machine, connection times between 15 seconds and nearly 2 minutes are typical using current hardware and algorithms. As keys take more computing power to generate, which is likely because the requirements of key size will increase, the exemplary illustrative non-limiting implementation will allow a quick logon time from a computationally limited calling party. In some client-server embodiments, a server may generate keys constantly (i.e., semi-dynamically). Semi-dynamic key generation in a client-

server embodiment allows connection latency to be independent of client and server hardware requirements.

[0066] Still another benefit of the exemplary illustrative non-limiting technology herein is that human users don't need to bring anything with them when connecting to another party in a peer-to-peer embodiment or logging on in a client-server embodiment. All that a human user needs to supply is a password. There is no need for a human user to carry a floppy disk with his or her stored "authorized key pair," which reduces the potential for human error, e.g., an administrator emailing a key pair to a human user. Also, there is no need for a human user to have a physical token to log on. This enables true mobility by preventing the need for hardware such as a SmartCard reader.

[0067] A security benefit of the exemplary illustrative non-limiting technology herein comes from the double integrity of the shared secret key. In an exemplary illustrative non-limiting implementation, the shared secret key is not sent in a single message. The shared secret key is computed based on the values of two shared random numbers. The first shared random number is encoded using a function based on the password. The second shared random number is encoded using a random public key. So even if someone were able to somehow steal a user's password, he still would have no way to decrypt the messages that the user had transmitted in the past. Secure data remains secure.

[0068] The double integrity of the shared secret key adds another benefit: if an attacker is able to penetrate a party and manipulate it prior to or during a communications session, it might in this way be able to weaken, sabotage, specify, or intercept a shared random number. However, without the other shared random number, the attack would not yield the shared secret key. No attack that targets only one shared random number can weaken the shared secret key.

[0069] The exemplary illustrative non-limiting technology herein provides strong security protection in obtaining a shared secret key. The following are some

exemplary illustrative non-limiting examples of attacks that are thwarted by the technology herein:

- No password eavesdropping
- No form of the password is ever sent, so one could never obtain the password.
- No password database hijacking
- The password database itself is encrypted, so one would never be able to grab any form of the password, to launch an off-line password guessing attack.
- No reflection (man-in-the-middle) attack
- At no point can a message be pulled from one authentication session, and used sensibly in a different session, to obtain access to keys or messages.
- No replay attack
- Because of the use of timestamps in some exemplary illustrative non-limiting embodiments , it is not possible to use untampered messages to cause either party to undergo processing that would ordinarily prohibit a valid user from gaining access to the system.
- No impersonating called party
- Impersonating the server would never gain an impersonator access to a password, because no form of the password is ever sent by the calling party. (The called party uses a function of the password as a key.). For this matter, it would never make sense for an attacker to impersonate the called party.

[0070] The specific algorithms and steps described herein, as well as the basic steps which such algorithms represent (even if they are replaced by different algorithms), are designed for implementation using general purpose microprocessors. Furthermore, each of the algorithms and steps described herein,

as well as the basic steps represented by such algorithms, can be encoded on computer storage media such as CD ROMS, floppy disks, computer hard drives, and other magnetic, optical, other machine readable media, whether alone or in combination with one or more of the algorithms and steps described herein.

[0071] Although the methods discussed herein have been described in detail with regard to some exemplary embodiments and drawings thereof, it should be apparent to those skilled in the art that various adaptations and modifications of the methods can be accomplished. Thus, by way of example and not of limitation, the methods are discussed as illustrated by the figures. Accordingly, the invention is not limited to the precise embodiments shown in the drawings and described in detail hereinabove, but is set out in the following claims.